2010 IEEE International Conference on GLOBAL SOFTWARE ENGINEERING

23-26 August 2010 • Princeton, New Jersey, USA

CONFERENCE INFORMATION

PAPERS BY SESSION

PAPERS BY AUTHOR

GETTING STARTED

TRADEMARKS

SEARCH

Sponsored by

IEEE

Published by

IEEE computer society

CPS Conference Publishing Services

IEEE Computer Society Order Number P4122
ISBN-13: 978-0-7695-4122-8
BMS Part # CFP10ICG-PRT
Library of Congress Number 2010927211

Additional copies may be ordered from:

IEEE Computer Society
Customer Service Center
10662 Los Vaqueros Circle
P.O. Box 3014
Los Alamitos, CA 90720-1314
Tel: + 1 800 272 6657
Fax: + 1 714 821 4641
http://computer.org/cspress
csbooks@computer.org

IEEE Service Center
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
Tel: + 1 732 981 0060
Fax: + 1 732 981 9667
http://shop.ieee.org/store/
customer-service@ieee.org

IEEE Computer Society
Asia/Pacific Office
Watanabe Bldg., 1-4-2
Minami-Aoyama
Minato-ku, Tokyo 107-0062
JAPAN
Tel: + 81 3 3408 3118
Fax: + 81 3 3408 3553
tokyo.ofc@computer.org

Individual paper REPRINTS may be ordered at: <reprints@computer.org>

Editorial production by Patrick Kellenberger
Cover art production by Mark Bartosik
Printed in the United Stated of America by Applied Digital Imaging

IEEE Computer Society
Conference Publishing Services **(CPS)**
http://www.computer.org/cps

Welcome Message
ICGSE 2010

The 5th International Conference on Global Software Engineering (ICGSE) brings together researchers and practitioners interested in exploring how globally distributed teams work and how the challenges posed by global software engineering can be met. This conference is a forum at the intersection of software engineering, communications, collaboration, business, and cultural aspects that influence human behaviors when faced with software development in global environments. This is the fifth in an annual series of international conferences on Global Software Engineering that began almost five years ago in Florianópolis, Brazil, in 2006. In our subsequent sessions we have learned from different cultures, where global software engineering is thriving: Germany, India, Ireland, and in our fifth edition: North-America.

The research agenda presented at the previous meetings has influenced the research field in global software engineering. We notice more formal modeling and empirical studies, and increased focus on the education of the new software engineers.

This year's technical program is as strong as ever. We received 55 submissions which were thoroughly reviewed by three or four reviewers from an expert program committee. From these 23 were accepted as research papers, 4 as industrial experience papers and 4 as educational papers. The Conference is structured in 9 sessions over 3 days in a single track. Our technical program attempts to address this broad area by presenting new insights into new tools, management, processes, human aspects, and teaching as applied to global software engineering.

The conference provides great opportunities for open discussion of issues and research directions, prompted by three workshops (PARIS'10, Knowing, REMIDI) and one panel. Our panel topic should provide an opportunity for us to travel virtually into the future and to speculate on how the state of the art in communication technologies will likely impact the workplace, specifically with the modern advances in communication technologies.

We offer our sincere thanks to the many individuals and organizations that helped make this year's Conference possible: the IEEE Computer Society, Siemens AG, Siemens Corporate Research, the ICGSE steering committee, the ICGSE 2010 program committee, and the ICGSE 2010 organizing committee. Finally, we would like to thank the authors, the tutorial and keynote speakers, and the workshop organizers and participants for making this year's ICGSE an exciting event.

Welcome to ICGSE 2010 in Princeton and enjoy the Fifth International Conference on Global Software Engineering!

August 2010

Alberto Avritzer, General Chair
Yael Dubinsky, Program Co-chair
Allen Milewski, Program Co-chairs

2010 International Conference on Global Software Engineering

ICGSE 2010

Table of Contents

Industry Experiences

Human Aspects

Preparing Software Engineers for Global Software Environments

Management Environments II: Assessing the Risk

Tools II: Emerging Tools

Management Environments III: Understanding Effort in a Global Context

PARIS'10: Methods and Tools for Project/Architecture/Risk Management in Globally Distributed Software Development Projects

REMIDI'10: Fourth International Workshop on Toll Support Development and Management in Distributed Software Projects

KNOWING'10: Knowledge Engineering in Global Software Development

Doctoral Symposium

2010 International Conference on Global Software Engineering

Preparing Students and Engineers for Global Software Development: A Systematic Review

Miguel J. Monasor
University of Castilla-La Mancha
Campus Universitario s/n, 02071, Albacete, Spain
MiguelJ.Monasor@gmail.com

Aurora Vizcaíno, Mario Piattini, Ismael Caballero
Alarcos Research Group, Institute of Information Technologies & Systems
Escuela Superior de Informática
University of Castilla-La Mancha
Paseo de la Universidad 4, 13071, Ciudad Real, Spain
{Aurora.Vizcaino, Mario.Piattini, Ismael.Caballero}@uclm.es

***Abstract*— In recent years, the evolution of Global Software Development (GSD) has grown both rapidly and significantly, and although the efficiency of this new type of development has been proven, some challenging issues must still be confronted. Of all these, our research line is focused on designing the specific training that members of virtual teams must receive. Universities and companies therefore need to design training schemas to deal with the specifics of GSD, which are principally related to communication difficulties and time and cultural differences.**

In this work we present the findings of a Systematic Literature Review in the field of GSD training and teaching. Our intention is twofold: on the one hand we wish to discover the existing strategies and proposals available up to the present day, and on the other hand we wish to identify the open challenges, that will be helpful for practitioners and researchers in the future.

Keywords-global software development; distributed software development; teaching; education, training, learning, systematic literature review

I. INTRODUCTION

Global Software Development (GSD) is an emerging paradigm in which development teams are geographically distributed whilst working on the same projects [1]. The main reason for this shift is to optimize and decrease costs by finding zones in which a skilled workforce is more readily available.

However, this type of development also entails certain drawbacks, caused mainly by distance, time and cultural and language differences, which make it difficult to reach a common understanding, particularly when members use non-native languages and employ terms that may be misinterpreted [2], [3]. Traditional face-to-face meetings are no longer common, making communication and coordination more complex. In addition, the interaction between members requires the use of technology [4], which introduces more variables leading to a minimization in the effects of the well-known advantages of this shift.

These factors influence the way in which software is defined, built, tested and delivered to customers, thus affecting the corresponding stages of the software life cycle. Students and software engineers must therefore acquire the skills needed to confront the new challenges that are present in these environments, and which are not at present part of their conventional curriculum, since current software engineering education rarely takes GSD activities into account [5], [6]. Moreover, preparing students in this field is not easy, as it requires new theoretical contents and tools, along with a great deal of coordination.

In this work, we present the findings of a Systematic Literature Review (SLR) in the field of GSD training and education in order to permit researchers, instructors and practitioners to discover the main challenges, strategies and proposals available up to the present day.

This paper is organized as follows: Section II justifies the need for this SLR, Section III describes the SLR procedure used and the results that were obtained. Section IV presents an analysis of the results previously presented. The main proposals found are explained in Section V. In Section VI we explain the skills required for learners and instructors. The main success factors for carrying out the learning process are listed in Section VII. Finally, Section VIII provides some concluding remarks.

II. BACKGROUND AND MOTIVATION

Although GSD allows companies to take advantage of the higher availability of a qualified workforce in decentralized zones, their managers frequently complain that recent graduates lack the necessary skills to tackle the new problems created by GSD. They argue that their experience is strictly limited to relatively short projects, because education programs do not deal with these subjects at an appropriate level in general [7], and for GSD in particular.

On the other hand, training these skills is not easy, since it necessitates providing students and inexpert software engineers with real experiences that will allows them to develop both technical and non-technical skills [8]. In addition, companies are not always willing to invest their resources in training programs, for two main reasons: firstly, because some organizational resources must be relocated,

978-0-7695-4122-8/10 $26.00 © 2010 IEEE
DOI 10.1109/ICGSE.2010.28

putting real on-going projects at risk, and secondly, because reproducing the complexity of real settings is difficult to achieve in educational environments [9], and consequently training might not be successful. If companies were prepared to support this kind of training, instructors would have appropriate tools, methods and course materials with which to teach these skills at their disposal [10], thus providing learners with real experiences and case studies adjusted to the reality of the companies' requirements that would connect theory with practice [11].

Literature shows that several proposals have attempted to tackle this subject: academic courses [12], learning environments [10] and the application of tools presented in real scenarios [11].

We have therefore decided to make an in-depth study of the subject of training in GSD education by exploring the challenges and proposals that might be of great assistance to researchers, instructors and practitioners, and which will serve to define an educational environment in the future, based on rigorous findings of the state-of-the-art.

Before presenting our SLR we should like mention that various interesting SLRs on GSD exist such as that of Smite et al. [13], which is focused on empirical evidence in GSD and identifies some best practices in this still immature field. A further SLR in this area, which uses the Scrum methodology, is presented by Hossain et al. [14] and Khan et al. [15] and focuses on software outsourcing activities. However we have not found any SLRs in the particular field of GSD on which our work is focused.

III. Systematic Review Procedure

An SLR permits the identification, evaluation and interpretation of all of the available relevant studies related to a particular research question, topic area or phenomenon, according to a predefined strategy.

In this work, we have carried out an SLR by following the guidelines provided by Kitchenham and Charters [16]. The elements involved in the process are detailed in the following subsections.

A. Question Formularization

The research question that guided this SLR was:

What are the initiatives carried out in relation to Global Software Development training and education?

The ultimate goal of this SLR is to identify the best procedures, models and strategies employed in the training and education of software engineers. The population of primary studies will also be composed of publications that have been found in the selected sources. These publications deal with the skills required in GSD, and how to deal with the main problems identified in teaching these skills.

B. Sources Selection

We carried out an initial scoping study to determine the search string and the resources to be searched. The search string was established as follows:

("distributed software development" OR "global software development" OR "global software engineering" OR "distributed software engineering") AND ("learning" OR "teaching" OR "education" OR "training" OR "simulation" OR "simulator").

Our search strategy consisted of the following decisions:

- **Search sources**: We adapted the search chain to the search engines we have used, namely:
  - Science@Direct (www.sciencedirect.com)
  - SpringerLink (www.springerlink.com)
  - IEEE Digital Library (www.computer.org)
  - ACM Digital Library (http://portal.acm.org/dl.cfm)
  - Wiley Interscience (www.interscience.wiley.com)
  - AIS eLibrary (http://aisel.aisnet.org)

All the proceedings of the International Conference on Global Software Engineering (ICGSE) were taken into account as they appear in the IEEE Digital Library.

- **Language**: We only considered papers published in English.
- **Searched items**: Conference papers, journal articles and workshop papers.
- **Publication period**: The year 2000 until November 15th, 2009. Since, as is mentioned in [13], and stated by [17], GSD is a 21st century trend, it is during the last few years that most papers have been written about this field.

C. Relevant Studies Selection

The inclusion criteria for determining whether a study should be considered relevant (a potential candidate to become a primary study) was based on analyzing the title, abstract and keywords from the studies retrieved by the search to determine whether they dealt with initiatives related to GSD training and education and whether they made any proposals concerning GSD coordination, collaboration and communication problems. In some cases it was necessary to read the entire document to determine its precise scope and relevance.

After analyzing the first iteration of the SLR, we applied exclusion criteria to obtain the primary studies, excluding those studies whose orientation towards GSD problems was not well defined and did not contribute with any significant proposals, despite addressing the training or education of GSD.

D. Primary Studies Selection

The inclusion criteria for the selection of primary studies are listed below:

- Studies that describe GSD courses in university environments or companies.
- Studies that describe real experiences, problems or success factors.
- Studies that propose training tools or environments to carry out the training or education process.

We excluded those studies that fulfilled these criteria:

- Studies that do not answer the research question.
- Studies that do not contribute with any relevant information or proposals in GSD training or education.
- Studies that do not describe their methods to an appropriate level of detail.

E. Quality Assessment

After the initial selection of primary studies, we carried out the quality assessment of the studies in two stages. First we reviewed the appropriateness of the studies, bearing in mind the inclusion and exclusion criteria, and after we carried out a more detailed review in parallel with the information extraction process.

During this process, we verified the relevance and quality of the studies, also bearing in mind the clarity of their methods and proposals. We also excluded a number of papers that were published in different sources but were based on the same study, and versions of previously published papers, in order to avoid their repetition during the data extraction process. With regard to this, we excluded the studies listed in Table I:

TABLE I. STUDIES EXCLUDED FOR SIMILARITY

Excluded studies	*Similar to*
[18]	[19]
[20]	[21]
[22]	[6]
[23]	[6]
[24]	[25]
[26]	[28]
[27]	[29]

The search procedure (see Figure 1) produced 67 relevant studies. Of these, 38 were selected as primary studies. The complete list of primary studies is shown in Table II.

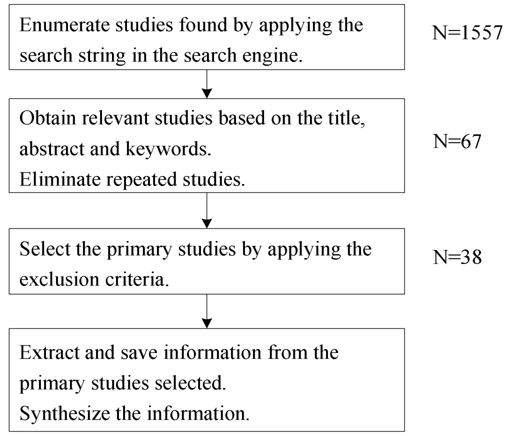

Figure 1. Selection process for primary studies

F. Information Extraction

We applied an information extraction process for each primary study by using a database with a pre-defined data extraction form containing the following information: title, authors, reference, year, researchers' country, source, number of pages, scope, proposal, organization type (university, company) and company size, processes covered, target population, date of review and methodology.

This process was initially carried out by a novice researcher. A more experienced researcher then reviewed the results in order to ensure that the selected studies matched the research question at a sufficient level of quality for them to be considered as primary studies.

TABLE II. LIST OF PRIMARY STUDIES

Reference	**Year**	**Source**	**Methodology**	**Reference**	**Year**	**Source**	**Methodology**
[30]	2000	IEEE	Case study	[31]	2007	IEEE	Case study
[32]	2001	IEEE	Experimental	[33]	2007	ACM	Case study
[34]	2002	IEEE	Non-experimental	[35]	2007	ACM	Case study
[10]	2002	IEEE	Case study	[36]	2005	IEEE	Case study
[37]	2005	IEEE	Non-experimental	[21]	2008	SpringerLink	Non-experimental
[38]	2008	IEEE	Experimental, Survey	[39]	2008	SpringerLink	Case study, Simulation
[40]	2005	ACM	Case study	[41]	2008	IEEE	Case study
[42]	2006	IEEE	Non-experimental	[6]	2008	IEEE	Case study
[43]	2006	IEEE	Experimental	[12]	2008	ACM	Case study
[5]	2006	IEEE	Case study	[44]	2008	ACM	Case study
[45]	2006	IEEE	Case study	[46]	2009	SpringerLink	Case study
[19]	2006	IEEE	Case study	[25]	2009	SpringerLink	Case study
[47]	2006	IEEE	Case study	[48]	2009	IEEE	Survey
[28]	2007	Wiley	Simulation	[49]	2009	ACM	Case study
[50]	2009	ACM	Case study	[51]	2007	SpringerLink	Case study
[52]	2007	IEEE	Non-experimental	[11]	2009	ACM	Case study
[53]	2007	IEEE	Case study	[54]	2009	ACM	Case study
[8]	2007	IEEE	Case study	[29]	2009	ACM	Case study
[55]	2009	IEEE	Experimental	[56]	2009	IEEE	Case study

IV. TRENDS IN GSD EDUCATION RESEARCH

This section analyzes and discusses the content and characteristics of the primary papers found.

The methodology of the primary studies selected was categorized as [57]: case studies, literature reviews, experiments, simulations and surveys. The non-experimental model (for studies that make a proposal without testing it or performing experiments) was also applied.

Figure 2 shows that the majority are case studies which basically describe experiences in university courses. We did not find any SLR specifically concerning GSD education.

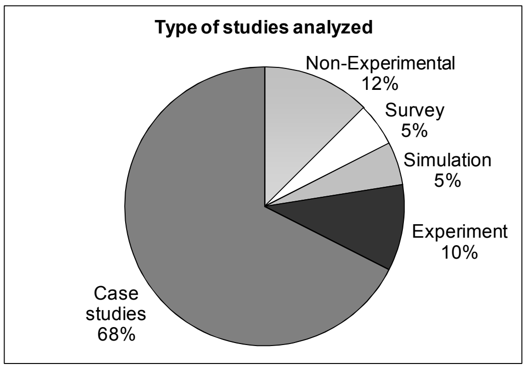

Figure 2. Type of studies analyzed.

Figure 3 shows that most of the primary studies are contextualized in a university environment. They describe how groups of students at different locations have carried out joint developments. It is interesting to note that we have also found companies' approaches and studies developed in collaboration between universities and companies.

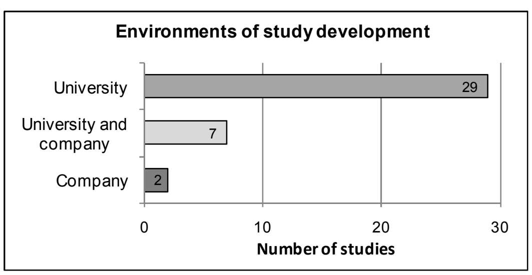

Figure 3. Environments of study development.

A. Classification of Primary Studies according to ISO/IEC 12207

In order to discover the main areas in which studies have focused to date, the primary studies were classified according to the studied processes of the software life cycle, based on the ISO/IEC 12207 standard [58]. The results of the mapping are presented in Table III. Some primary studies were not classified since their scope was more generic.

Upon considering the number of publications associated with each process category, it is possible to state that the greatest efforts are mainly focused on software construction, software design, requirements engineering and software testing. From the point of view of specific characteristics of GSD, it is possible to conclude that all these processes are particularly affected by communication problems and cultural and language differences. This is therefore consistent with our initial ideas that the training and teaching of these skills have become an important topic. Some of the studies included dealt with more than one process at a different level of detail.

TABLE III. PROCESSES CONSIDERED IN THE PRIMARY STUDIES

Process	*Studies*
Software construction	[49], [11], [30], [5], [44], [19], [43], [8], [47], [10], [35], [48], [6]
Software design	[49], [30], [5], [19], [43], [8], [47], [10], [35], [6], [31], [38]
Requirements analysis	[49], [30], [5], [19], [43], [8], [10], [6]
Software testing	[49], [50], [30], [19], [47], [10], [52]
Project management	[30], [12], [19], [43], [10], [35], [46]
Requirements elicitation	[49], [30], [5], [19], [21], [6]
Organizational management	[34], [43], [51], [39], [28]
Configuration management	[11], [44], [8], [31]
Quality Assurance	[12], [44], [19], [25]
Documentation	[49], [44], [35]
Problems resolution	[11]
Change requests	[11]
Joint reviews	[12]
Management process	[51]
Risk management	[35]
Software integration	[56]

B. Publications Tendency

Upon considering the number of primary studies found published by year, the subject of teaching GSD is evidently an area which was not widely studied until a few years ago and, as Figure 4 shows, 2009 is the year in which the greatest number of studies were published, bearing in mind that the search was completed in November of that year.

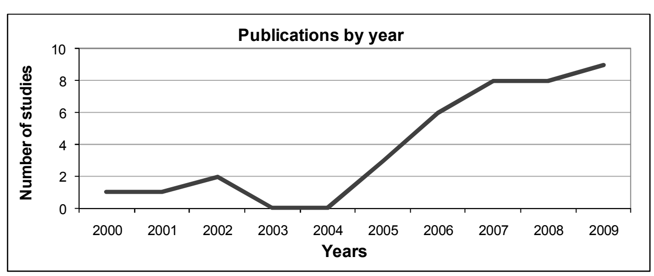

Figure 4. Trends in publications in GSD education.

Table IV shows the origin of the authors of each paper considering their affiliation, who are largely from the USA.

TABLE IV. CONTRIBUTORS ORIGIN

Country	*Contributors*	*Country*	*Contributors*
USA	26	Canada	1
Germany	6	Australia	1
India	5	Chile	1
Cambodia	3	Japan	1
UK	3	Mexico	1
Ireland	2	Puerto Rico	1
Italy	2	Spain	1
Netherlands	2	Sweden	1
Panamá	2	Thailand	1
Turkey	2	Brazil	1
Austria	1		

V. FINDINGS FROM THE SLR

In this section, we synthesize the challenges, methods and proposals identified through the SLR, discussing the most relevant studies. We have structured this section according to the following types of proposals found:

-Learning environments
-e-Learning approaches
-Simulators
-Teaching GSD in the classroom
-Training GSD in the classroom
-Teaching GSD in the company

A. *Learning environments*

We found some primary studies dealing with learning environments that provide functionalities to manage the training of GSD activities and which can be used in companies or universities.

iBistro [34] is an augmented space based on the 'learning by doing' approach, thus enabling distributed members to collaborate in the development of software that can be used to learn project management, software development and social skills.

This environment addresses miscommunications and information problems in informal meetings by allowing students to capture structures and retrieve knowledge from the meetings by using the audio, video, sketches, notes and the drawings generated. This is supported by a minute generator tool that stores the contextual information, allows the meetings to be represented and facilitates navigation through the database which contains information about the meeting.

iBistro also provides intelligent support mechanisms with which to perform certain tasks such as computer supported group formation, and the ability to effectively find stakeholders and experts in certain areas.

We have also found blended learning approaches, such as that presented in [40] for a lab course on Distributed Software Development, based on the collaborative virtual learning environment CURE [59]. This lab course uses virtual places (called rooms) for collaboration. These virtual places may contain pages (content), communication channels (such as chat, threaded mailbox, etc.), and users, who will interact with other users standing in the same room.

This proposal is based on the Problem-Based Learning approach [60], in which, students have to form groups and are encouraged to find a solution to a problem in a collaborative manner. During the problem resolution, they submit deliverables at prescribed milestones.

On the other hand, in [36] a platform based on Eclipse is presented. This platform consists of the integration of CURE and CodeBeamer. CodeBeamer is a collaborative platform that offers integrated support in project management, requirements management and code management and allows asynchronous communication by means of a wiki system. The proposed platform allows students to collaborate during all the phases of the development to produce a large software system.

[51] is also focused on CodeBeamer. In this case, a framework is proposed which is oriented towards offshoring practices by expanding the functionality of this environment. The authors have basically developed the following Requirements Engineering oriented tools:

- Ibere: an internet-based tool that guides distributed members in the selection of requirements and the estimation procedure.
- TraVis: for traceability management and supporting change management and the visualization and analysis of the dependences among artefacts. This tool helps CodeBeamer to improve awareness by managing a wide range of information through graphical representations and real-time analysis of the project.

A Web-based collaboration platform that eases communication and content management, providing a discussion board, a file sharing repository and a project calendar is presented in [29]. Instructors can add training modules, and students can access their description along with their instructions, milestones, and deliverables. The students use this platform to work with their partners in order to achieve the module's scopes.

Jazz [11] is a collaborative development platform that integrates several functionalities to support the software life cycle, such as: source code repository, chat, web interface, reports generation, and work items. Students can generate work items containing all of the relevant information related to the resolution of a problem along with the associated chat conversations.

In [46], the authors detail a tool that supports problem analysis and helps students to improve their problem-solving skills. Users can add labels to problem nodes of the project that will be helpful in the problem resolution.

[47] describes a computer supported collaborative tool for teaching distributed teams through the use of collaborative tools (including chat, a scribble tool, an application sharing tool, graphics tools for designing UML documents, etc.). The authors have also developed course management software that helps instructors in tasks related to the administration of groups and the collection of information concerning the student's actions, thus providing a means to evaluate them.

This study also presents a Web portal developed with the aim of helping students to manage the groups and projects in which they are involved. Students use this means to share their personal information (such as name or email address) with their partners and they can also access their partner's schedule in order to agree possible meeting times for the projects they have in common.

[32] proposes a framework which is useful for dealing with some of the difficulties in GSD. It basically consists of the following set of tools:

- Project scheduling and tracking tool: to help the instructors in their tasks.
- Configuration management tool: that manages change requests by filling out and submitting a form.
- Technical review tool: including inspections and walkthroughs.

This study also introduces the idea of the **team contract**, in order to standardize the project's daily process and avoid missed meetings and other problems caused by the former informality of the process. The team drafts a contract to establish rules concerning communication, response times and the authority that is allowed to manage these kind problems. This same idea is also used in [29].

B. e-Learning approaches

On the other hand we have also found the application of e-learning approaches which, in contrast to learning environments, consists of web-based applications oriented towards delivering online courses.

OAS!S [54] is a virtual learning environment created by customizing the WebCT Vista / Blackboard platform. Its use is broadly extended in many universities. It allows discussion boards, mail systems, chat and content management, which can be very helpful, mainly in improving communicative skills.

In this respect we should also mention the open source platform learning OLAT (Online Learning And Training) [54], that similarly supports forums, chats, file sharing, and mailing system, offering support for various e-learning standards such as IMS or SCORM.

Finally, [46] presents the development of a Web-based e-learning course dealing with project integration and quality management (including project chartering, monitoring, and controlling), along with time and cost management through scheduling tasks and activities using methods for controlling their cost.

C. Simulators

This SLR has led us to realize that some primary studies propose simulators which permit the training of specific skills, with the advantage of reducing the costs and risks of performing that task if it were to be performed in real environments.

In this respect, [28] proposes a simulator to study different ways in which to configure GSD projects. The configuration includes parameters related to task allocation, studying phase-based, module-based and follow-the-sun allocation strategies. The simulator computes the project duration for each configuration, thus allowing the student to study the impact of different GSD factors, such as distance, culture, language, trust and time zone, on project duration. This simulator could therefore be used not only to help managers to find the best configuration settings in order to improve their project performance, but also as a training device in the new challenges created by GSD, by helping managers to better estimate the values of the parameters that make a project suitable for GSD, or which sites should be included in the project, how the work should be divided, or which tools are effective in these environments.

We found another approach in [21] regarding the simulation of the Requirements Engineering (RE) process. RE has become one of the most problematic processes in GSD since it requires a great deal of communication and interaction with multicultural and multidisciplinary members. Requirements engineers must assimilate new methods and must also be trained in appropriate technologies which allow them to tackle the new challenges created *by GSD.*

This simulator allows students and professionals to be trained in the skills needed in the elicitation of requirements interviews. Learners are placed in a simulated interview in which they interact with virtual humans coming from different cultures in order to obtain the functional and non-functional requirements of a project. They therefore learn computer-mediated communication knowledge and attain a higher understanding of the cultures and customs of other countries.

D. Teaching GSD in the classroom

Traditional theoretical classes are also commonly found in literature as a response to the needs for adjustments in software engineering education [31]. In this respect, many studies coincide in highlighting the necessity for joint courses with different universities, so that students can benefit from a wide set of knowledge and experiences [44]. However, a commonly reported problem with this approach is related to the difficulty of attaining an appropriate level of coordination and collaboration with the different institutions [42].

We have found several studies that describe their learning course approaches, such as [30], which presents an experience in collaboration with other universities with three distributed software engineering project courses. The approach not only includes theoretical classes, but also seminars and laboratories through a collaboration infrastructure.

A European Masters program on Global Software Engineering which involved several universities from different countries was presented in [31]. This program enables the technical and cultural dimensions of GSD to be taught and is focused on the areas of: software architecting, real-time embedded systems engineering and web systems and services engineering.

[37] provides details of a software engineering program to incorporate collaborative global software development processes into universities and industrial software companies and also to enhance distant learning programs.

The Masters course proposed in [38] takes into account the fact that students from different universities have different backgrounds, skills and experience, so it is important to analyze their prior education and characterize the students in order to design the course.

As we shall show in the following section, some studies also combine teaching classes with practices and actual developments.

E. Training GSD in the classroom

"Learning by doing" is the most common approach suggested by the primary studies since developing the GSD skills requires putting theory into practice by using tools and methods for tackling typical problems found in real environments.

Universities that teach GSD tend to organize joint student developments, collaborating with universities from different countries. In these cases, the students communicate by using email, telephone and instant messaging [33] and this interaction allows them to learn from others students' skills and cultures. Their involvement in real development experiences is therefore both positive and beneficial for their curriculums [8], given that they tackle processes with a close similarity to those applied in industry [10].

In [10], we found an experience in which several universities from the same country participated. Distant team members collaborated by documenting each task and sending it electronically to the other university groups. The process defined was flexible enough to allow modification in the school's task assignments.

[49] present a case study of a course in which virtual teams from different universities collaborated to develop a project for a real customer that profited from this learning process. However this study is limited to the interaction between only two universities, both of which were located in the same country, so they could not deal with all the problems caused by internationalization with regard to language, time and cultural problems.

The experience presented in [43], in which three universities from different countries were involved, identifies scheduling as one of the most common difficulties for engineers when they are full-time developers. One of the important subjects mentioned in this study consists of training communication and informal skills and learning how to work effectively with a team and react quickly to changes in requirements, architecture and organization.

[19] combine a class which teaches software engineering methods and processes with the development of a real project in a setting designed to simulate a small company. Students were divided into small groups (in some cases distributed throughout different countries), and had to collaborate in order to develop a complete working application.

In [54] two projects involving students from four countries were developed with the aim of studying a team performance model that was used to measure certain factors that affect collaborative work.

In [25], the authors present a course on quality assurance from three universities from different countries that have worked together on the development of a software system. This practice allowed students to interact with partners of different cultures, and also to play different roles in the project. Since they collaborated with partners from different backgrounds, skills and ways of thinking, their learning experience was richer that it might otherwise have been. This study, complemented in [6], makes some recommendations oriented towards project planning in GSD education based on the students' experiences. The interaction was carried out by using several types of communication tools [6]:

- Mailing lists, emails and chats.
- Wikis: Each sub-team in the project maintained a wiki containing all the documents and artifacts that they produced.
- Blogs: in which the students reflected their weekly progress in the project.

The authors also suggest that it is easier for course instructors to play the role of project managers. This permits them to take advantage of their position in order to lead students through the processes, and for instance, prevent them from experiencing certain coordination problems.

In [5], the corresponding authors report the strategies applied in a course on requirements engineering in three different universities, in which a web-based inspection tool named IBIS was used in an educational environment. These strategies were oriented towards learning the skills needed for international teamwork, along with the use of specific tools for remote communication.

Finally, we should also mention some outsourcing experiences for students reported in [33], courses oriented towards eXtreme Programming in distributed environments [12], and other experiences referring to offshore software development between undergraduate students at different locations replicating a client/vendor relationship in a virtual setting, as presented in [35].

F. Teaching GSD in the enterprise

Although the topic of teaching GSD within a company has not been commonly reported in the primary studies, we have found some related experiences. The most remarkable example was found in [41], which presents a training initiative in a multinational organization that applies GSD. A training course related to concepts and best practices of communication, trust, cultural differences and coordination was developed. After the completion of each project, the students generated a document containing the lessons learned.

The higher level of availability of an experienced workforce appearing in these environments makes the application of the concept of **learning networks** [25] possible. Since instructors cannot be experts in all the GSD areas and cannot consequently cover every topic, learning networks provide a new way in which to perform the learning process based on the experience of a multidisciplinary set of trainers.

An example of a learning network is described in [53], It consists of a team of trainers from a company who were experts in specific software development activities and who, in this case, combined their training activities with their work as engineers, project managers, quality managers, etc., in that company. Students were therefore able to interact with them and take advantage of real work experiences by maintaining contact with professionals who were carrying out day-to-day work.

VI. Required Skills in GSD

In this section we list the skills that learners should develop according to the findings of this SLR, along with the desirable skills required by the instructors oriented towards the challenges created by GSD.

A. Learners skills

We found the following main skills that learners should develop:

- Computer-mediated communications [29], [8], [5].
- Knowledge of communication protocols and customs [8], [6].
- Ability to communicate effectively using a common terminology and language [46].
- Knowledge of negotiation skills and contract writing in a common language [5].
- Managing ambiguity and uncertainty [8].
- Use of knowledge and document management and control version tools [34], [35].
- Leadership skills, and time management skills [53], [19].
- Skills to gain the team's confidence and trust [46].
- Ability to think from the perspective of the other side, teamwork skills [32].
- Informal communication and improvisation skills [43].
- Conflict resolution [29], particularly affected by culture differences and communication problems.
- Knowledge of tools, methods, data, and processes required in a distributed project [41].
- Communication with a multidisciplinary team [10].

B. Instructors' skills

The set of skills required by the instructors who teach GSD foundations largely depends on the course orientation (theoretical, practical, degree of collaboration with other institutions, etc.). However, we found that instructors must manage similar skills to those required by learners. This is owing to the fact that they must also have a basic knowledge of the problems of each role in the GSD, since a profound knowledge in all the topics is unviable [53], but they must be familiar with the course objectives and methods [10].

They also require minimal technical skills since they need to maintain the tools and course materials on the Internet. In some cases, instructors will play the role of project managers, and in other cases they will have to advise team leaders to confront certain conflictive situations [25], so they must know how to coordinate the team and maintain the motivation level by promoting effective team work and trust within the group [8]. They must also mediate to solve personal differences among members, repair technology, and resolve communication problems [47].

In addition, course material and training modules must be integrated from a variety of sources, and the tools used should be specifically selected, so the course preparation is challenging for the instructor, and more so when they have to coordinate their efforts with other institutions [5].

Finally, instructors must know how to create groups by selecting the most appropriate members according to their skills and knowledge. They also need to guide students, monitor and evaluate their activities, control project scheduling [25] and manage technological risks [35].

VII. Desirable Features for a Tool for GSD

From the experimental studies analyzed, we have extracted a set of characteristics that an environment oriented towards the training and education of GSD should meet:

- Provide training in informal communication [43] and promote the learners' proactivity [25].
- Support the interaction of distributed teams by applying communication and collaboration technology [41], [37].
- Allow the teams' performance to be quantified [55].
- Provide opportunities for self-reflection and self-correction [35].
- Provide rationale and explain consequences by sharing corporate experiences [25].
- Allow students to play different roles in the project to make them aware of the different kind of problems [5], [25].
- Help to create a feeling of trust between the members [5], [29], [10], [32], facilitating the knowledge of team ethics [44].
- Support the development of real-world projects [51] involving distant members with different cultures [30].
- Minimize the cost of the tools and infrastructure needed, along with their maintenance effort [53].
- Minimize the need for adherence to schedules of distant members [29], [44].

VIII. Conclusions

In this work we have applied an SLR method in order to analyze the literature related to GSD training and education. The results obtained have allowed us to depict a vision of the challenging factors and strategies applied in the teaching and training of the new skills required. These results have led us to the following conclusions:

Conclusion 1. There is a growing interest in GSD training and education, since GSD has grown in recent years and training has become essential.

Conclusion 2. The teaching and training of GSD must be supported by practical experiences through which students can learn by doing.

Conclusion 3. Simulating the complexity of real environments is difficult for universities, and the different timetables of the students make it difficult to coordinate training projects.

Conclusion 4. It is not possible for instructors to cover all the stages and problems of GSD [53], so any initiative should be focused on a specific field.

Conclusion 5. Students involved in GSD training programs usually experience a lack of motivation, schedule problems and communication difficulties [40], and this is greater still when cultural and language differences are present [32].

Conclusion 6. Particular training scenarios and learning environments require specific tools for communication, collaboration and document management. An appropriate selection of tools is therefore a key aspect [5].

Our future work will be focused on the development of a training environment for GSD which will be capable of simulating meetings at different stages of the project life cycle and of considering the problems concerning distance and cultural differences. Moreover, we are planning to have a contact with industry in order to know what kind of skills managers demand in software engineers who will work in GSD.

The work presented here has served as a starting point from which to establish the main foundations and criteria to guide our research. It has also served to collect ideas to define the training environment. The information collected will also be helpful in designing suitable training scenarios.

Finally, as our search was reduced to studies which addressed the training and education of GSD, we believe that this study could be extended. Since GSD is a wide area, it is possible to find related approaches which, despite not addressing this topic directly, deal with training and education in related fields of software engineering, such as those related to language learning and cultural differences, signifying that their study would therefore be important in a future work.

Acknowledgments

This work has been funded by the PEGASO/MAGO project (Ministerio de Ciencia e Innovación MICINN and Fondos FEDER, TIN2009-13718-C02-01). It is also supported by MEVALHE (HITO-09-126) and ENGLOBAS (PII2I09-0147-8235), funded by Consejería de Educación y Ciencia (Junta de Comunidades de Castilla-La Mancha), and co-funded by Fondos FEDER, as well as MELISA (PAC08-0142-3315) and TALES (HITO-2009-14), Junta de Comunidades de Castilla-La Mancha, Consejería de Educación y Ciencia, and DQNET (TIN2008-04951-E) Ministerio de Ciencia e Innovación in Spain.

References

[1] J. D. Herbsleb and D. Moitra, "Global software development," IEEE Software, vol. 18, pp. 16-20, 2001.

[2] Y. Hsieh, "Culture and Shared Understanding in Distributed Requirements Engineering," presented at the IEEE International Conference on Global Software Engineering (ICGSE'06), Florianópolis, Brazil, 2006.

[3] B. Lutz, "Linguistic Challenges in Global Software Development: Lessons Learned in an International SW Development Division," presented at the Fourth IEEE International Conference on Global Software Engineering (ICGSE'09), Limerick, Ireland, 2009.

[4] D. Damian, et al., "Using Real-Time Conferencing Tools to Leverage the Collaborative Potential of Global Teams: An Experience Report from Siemens IT Solutions and Services," presented at the Fourth IEEE International Conference on Global Software Engineering (ICGSE'09), Limerick, Ireland, 2009.

[5] D. Damian, et al., "Instructional design and assessment strategies for teaching global software development: a framework," presented at the Proceedings of the 28th international conference on Software engineering, Shanghai, China, 2006.

[6] O. Gotel, et al., "Working Across Borders: Overcoming Culturally-Based Technology Challenges in Student Global Software Development," presented at the Proceedings of the 2008 21st Conference on Software Engineering Education and Training, 2008.

[7] H. Su, et al., "Providing an integrated software development environment for undergraduate software engineering courses," J. Comput. Small Coll., vol. 23, pp. 143-149, 2007.

[8] I. Richardson, et al., "Globalizing Software Development in the Local Classroom," presented at the Proceedings of the 20th Conference on Software Engineering Education & Training, 2007.

[9] C. Ghezzi and D. Mandrioli, "The challenges of software engineering education," presented at the Proceedings of the 27th international conference on Software engineering, St. Louis, MO, USA, 2005.

[10] L. J. Burnell, et al., "Teaching Distributed Multidisciplinary Software Development," IEEE Softw., vol. 19, pp. 86-93, 2002.

[11] A. Meneely and L. Williams, "On preparing students for distributed software development with a synchronous, collaborative development platform," presented at the Proceedings of the 40th ACM technical symposium on Computer science education, Chattanooga, TN, USA, 2009.

[12] C. Murphy, et al., "A distance learning approach to teaching eXtreme programming," SIGCSE Bull., vol. 40, pp. 199-203, 2008.

[13] D. Šmite, et al., "Empirical evidence in global software engineering: a systematic review," Empirical Software Engineering, vol. 15, pp. 91-118, 2010.

[14] E. Hossain, et al., "Using Scrum in Global Software Development: A Systematic Literature Review," presented at the Fourth IEEE International Conference on Global Software Engineering (ICGSE'09), Limerick, Ireland, 2009.

[15] S. U. Khan, et al., "Critical Success Factors for Offshore Software Development Outsourcing Vendors: A Systematic Literature Review," presented at the Fourth IEEE International Conference on Global Software Engineering (ICGSE'09), Limerick, Ireland, 2009.

[16] B. Kitchenham and S. Charters, "Guidelines for performing Systematic Literature Reviews in Software Engineering," Keele University and Durham University Joint Report9 July, 2007 2007.

[17] T. L. Friedman, The world is flat: brief history of the 21st century. New York: Farrar, Straus and Giroux, 2005.

[18] D. Petkovic, et al., "Teaching Practical Software Engineering and Global Software Engineering: Case Study and Recommendations," in Frontiers in Education Conference, 36th Annual, San Diego, CA, USA, 2006, pp. 19-24.

[19] D. Petkovic, et al., "Teaching practical software engineering and global software engineering: evaluation and comparison," SIGCSE Bull., vol. 38, pp. 294-298, 2006.

[20] M. Romero, et al., "A Simulator for Education and Training in Global Requirements Engineering: A Work in Progress," presented at the Proceedings of the 2008 Eighth IEEE International Conference on Advanced Learning Technologies, 2008.

[21] M. Romero, et al., "Using Virtual Agents for the Teaching of Requirements Elicitation in GSD," presented at the Proceedings of the 8th international conference on Intelligent Virtual Agents, Tokyo, Japan, 2008.

[22] O. Gotel, et al., "Quality-Driven Competition: Uniting Undergraduates, Graduates and Professionals on Global Software Development Projects," in 21st IEEE-CS Conference on Software Engineering Education and Training Workshop, 2008. CSEETW '08., Charleston, South Carolina, USA, 2008, pp. 19-21.

[23] O. Gotel, et al., "Introducing Global Supply Chains into Software Engineering Education," in Software Engineering Approaches for Offshore and Outsourced Development. vol. 4716, S. Berlin, Ed., ed Heidelberg, 2007, pp. 44-58.

[24] O. Gotel, et al., "A Global and Competition-Based Model for Fostering Technical and Soft Skills in Software Engineering Education," presented at the Proceedings of the 2009 22nd

Conference on Software Engineering Education and Training - Volume 00, 2009.
[25] O. Gotel, et al., "Students as Partners and Students as Mentors: An Educational Model for Quality Assurance in Global Software Development." vol. 16, S. Berlin, Ed., ed Heidelberg, 2009, pp. 90-106.
[26] S.-o. Setamanit, et al., "Exploring the Impact of Task Allocation Strategies for Global Software Development Using Simulation," in Software Process Change. vol. 3966, S. Berlin, Ed., ed Heidelberg, 2006, pp. 274-285.
[27] R. Ocker, et al., "Enhancing Learning Experiences in Partially Distributed Teams: Training Students to Work Effectively Across Distances," presented at the Proceedings of the 42nd Hawaii International Conference on System Sciences, 2009.
[28] S.-o. Setamanit, et al., "Using simulation to evaluate global software development task allocation strategies," Software Process: Improvement and Practice, vol. 12, pp. 491-503, 2007.
[29] R. Ocker, et al., "Training Students to Work Effectively in Partially Distributed Teams," Trans. Comput. Educ., vol. 9, pp. 1-24, 2009.
[30] B. Bmegge, et al., "Transatlantic project courses in a university environment," presented at the Proceedings of the Seventh Asia-Pacific Software Engineering Conference, 2000.
[31] P. Lago, et al., "Towards a European Master Programme on Global Software Engineering," presented at the Proceedings of the 20th Conference on Software Engineering Education & Training, 2007.
[32] J. Favela and F. Peña-Mora, "An Experience in Collaborative Software Engineering Education," IEEE Softw., vol. 18, pp. 47-53, 2001.
[33] W. L. Honig and T. Prasad, "A classroom outsourcing experience for software engineering learning," presented at the Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education, Dundee, Scotland, 2007.
[34] A. Braun, et al., "iBistro: A Learning Environment for Knowledge Construction in Distributed Software Engineering Courses," presented at the Proceedings of the Ninth Asia-Pacific Software Engineering Conference, 2002.
[35] M. Adya, et al., "Bringing global sourcing into the classroom: experiential learning via software development project," presented at the Proceedings of the 2007 ACM SIGMIS CPR conference on Computer personnel research: The global information technology workforce, St. Louis, Missouri, USA, 2007.
[36] P. Bouillon, et al., "Software Engineering Projects in Distant Teaching," presented at the Proceedings of the 18th Conference on Software Engineering Education & Training, 2005.
[37] X. Liu, "Collaborative global software development and education," in Computer Software and Applications Conference, 2005. COMPSAC 2005. 29th Annual International, Edinburgh, Scotland, 2005, p. 371.
[38] P. Lago, et al., "Developing a Course on Designing Software in Globally Distributed Teams," presented at the IEEE International Conference on Global Software Engineering (ICGSE'08), Bangalore, India, 2008.
[39] S.-o. Setamanit and D. Raffo, "Identifying Key Success Factors for Globally Distributed Software Development Project Using Simulation: A Case Study," in Making Globally Distributed Software Development a Success Story. vol. 5007, S. Berlin, Ed., ed Heidelberg, 2008, pp. 320-332.
[40] T. Schümmer, et al., "Teaching distributed software development with the project method," presented at the Proceedings of the 2005 conference on computer support for collaborative learning: learning 2005: the next 10 years!, Taipei, Taiwan, 2005.
[41] R. Prikladnicki and L. Pilatti, "Improving Contextual Skills in Global Software Engineering: A Corporate Training Experience," presented at the IEEE International Conference on Global Software Engineering (ICGSE'08), Bangalore, India, 2008.
[42] U. Bellur, "An Academic Perspective on Globalization in the Software Industry," presented at the Proceedings of the 30th Annual International Computer Software and Applications Conference, 2006.
[43] I. Richardson, et al., "Distributed development: an education perspective on the global studio project," presented at the Proceedings of the 28th international conference on Software engineering, Shanghai, China, 2006.
[44] D. Petkovic, et al., "Assessment and comparison of local and global SW engineering practices in a classroom setting," presented at the Proceedings of the 13th annual conference on Innovation and technology in computer science education, Madrid, Spain, 2008.
[45] S. I. Ahamed, "Model for Global Software Engineering Project Life Cycle and How to Use it in Classroom for Preparing Our Students for the Globalization," in Computer Software and Applications Conference, 2006. COMPSAC '06. 30th Annual International, Chicago, Illinois, USA, 2006, pp. 59-60.
[46] S. Toyoda, et al., "A Case Study on Project-Management Training-Support Tools for Japanese/Chinese/Indian Offshore Development Engineers," in Knowledge-Based Intelligent Information and Engineering Systems. vol. 4693, S. Berlin, Ed., ed Heidelberg, 2009, pp. 1222-1229.
[47] K. Swigger, et al., "Teaching Students How to Work in Global Software Development Environments," in International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2006. CollaborateCom 2006., Atlanta, GA, USA, 2006, pp. 1-7.
[48] N. R. Mead, et al., "A Study of the Impact on Students Understanding Cross Cultural Differences in Software Engineering Work," in 33rd Annual IEEE International Computer Software and Applications Conference, Seattle, WA, USA, 2009, pp. 644-645.
[49] A. Rusu, et al., "Academia-academia-industry collaborations on software engineering projects using local-remote teams," presented at the Proceedings of the 40th ACM technical symposium on Computer science education, Chattanooga, TN, USA, 2009.
[50] A. B. Bondi and J. P. Ros, "Experience with Training a Remotely Located Performance Test Team in a Quasi-agile Global Environment," presented at the Proceedings of the 2009 Fourth IEEE International Conference on Global Software Engineering - Volume 00, 2009.
[51] K. Berkling, et al., "Offshore Software Development: Transferring Research Findings into the Classroom," in Software Engineering Approaches for Offshore and Outsourced Development. vol. 4716, S. Berlin, Ed., ed Heidelberg, 2007, pp. 1-18.
[52] M. Hackett, "Building Effective Global Software Test Teams through Training," presented at the IEEE International Conference on Global Software Engineering (ICGSE'07), Munich, Germany, 2007.
[53] B. Lutz, "Training for Global Software Development in an International "Learning Network"," presented at the Proceedings of the International Conference on Global Software Engineering, 2007.
[54] K. Swigger, et al., "Structural factors that affect global software development learning team performance," presented at the Proceedings of the special interest group on management information system's 47th annual conference on Computer personnel research, Limerick, Ireland, 2009.
[55] K. Swigger, et al., "A Comparison of Team Performance Measures for Global Software Development Student Teams," presented at the Fourth IEEE International Conference on Global Software Engineering (ICGSE'09), Limerick, Ireland, 2009.
[56] O. Gotel, et al., "Integration Starts on Day One in Global Software Development Projects," presented at the IEEE International Conference on Global Software Engineering (ICGSE'08), Bangalore, India, 2008.
[57] V. Z. Marvin and R. W. Dolores, "Experimental Models for Validating Technology," Los Alamitos, CA, USA 0018-9162, 1998.
[58] ISO/IEC 12207:2002, ISO, 2002.
[59] J. M. Haake, et al., "Supporting Flexible Collaborative Distance Learning in the CURE Platform," presented at the Proceedings of the Proceedings of the 37th Annual Hawaii International Conference on System Sciences (HICSS'04) - Track 1 - Volume 1, 2004.
[60] D. R. Woods, Problem Based Learning: How to Get the Most from PBL. Hamilton, Ontario, USA: McMaster University, 1994.